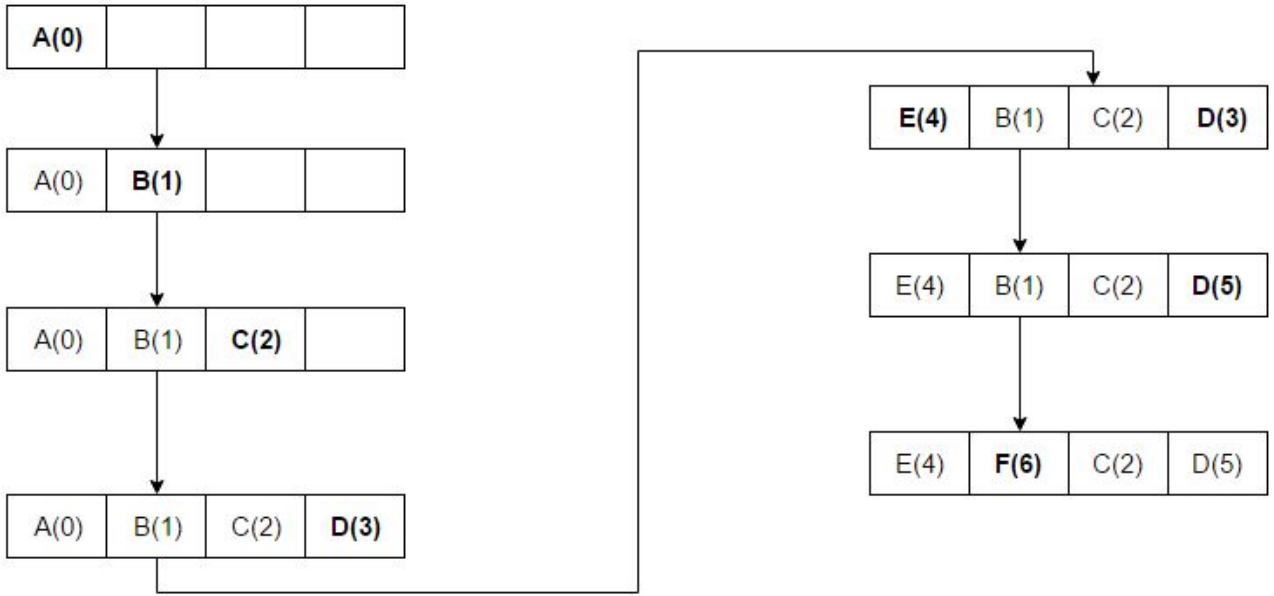
LRU vs MRU

CS448 Staff

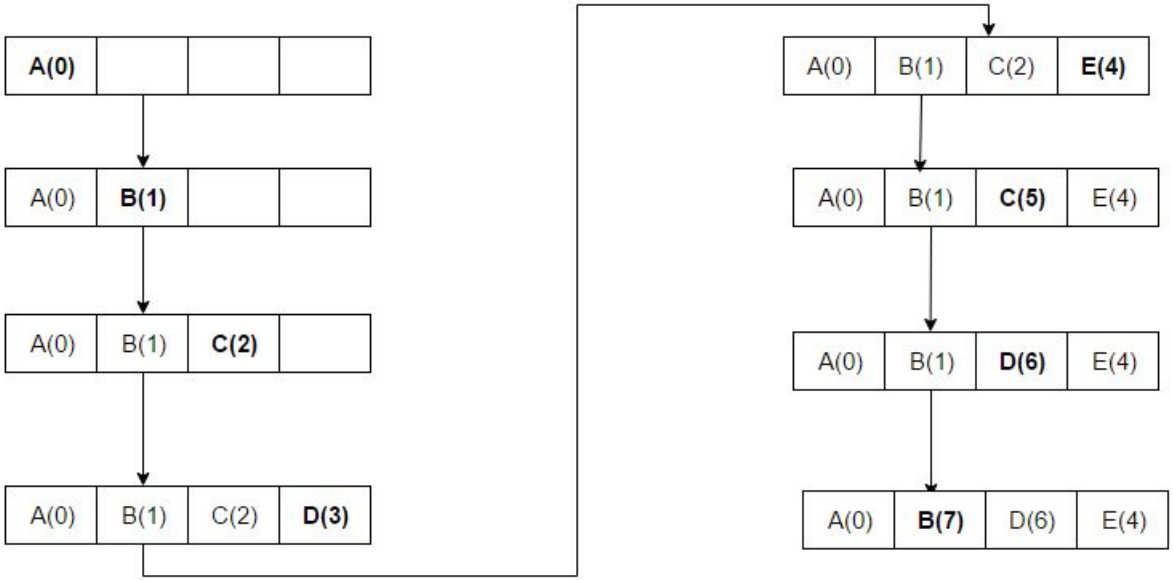
Least recently used (LRU)

- Discards the least recently used items first.



Most recently used (MRU)

- Discards, in contrast to LRU, the most recently used items first.



Sequential flooding : Nasty situation caused by LRU + repeated sequential scans.

- # frames < # pages in file.
- Using LRU, every scan of the file will result in reading every page of the file.
- Let us say there is 2 buffer frames Frame #1 and Frame #2, 3 pages in file P1, P2 and P3. What would happen if we scan the file twice(P1, P2, P3, P1, P2, P3) with sequential scan?

Sequential flooding (Continue) LRU

Block Read	Frame #1	Frame #2
P1	*P1*	
P2	P1	*P2*
P3	*P3*	P2
P1	P3	*P1*
P2	*P2*	P1
P3	P2	*P3*

As you can see, using **LRU**, every scan of a page will cost a page miss.
However, **MRU** will do much better.

Sequential flooding (Continue) MRU

Block Read	Frame #1	Frame #2
P1	*P1*	
P2	P1	*P2*
P3	P1	*P3*
P1	**P1 (Page hit)**	P3
P2	*P2*	P3
P3	P2	**P3 (Page Hit)**

Using **MRU**, We have two Page hits. In general, we can have #frame page hits in 2nd scan

Nested Join

- Suppose we have two #frames. Join two files, and each of them has 3 pages
  - for each record A do

    for each record of B do

    If record match then output ...

Nested Join: Smarter - compare everything in the pages

- Suppose we have two #frames. Join two files, and each of them has 3 pages
  - for each page PA of A do

    for each page PB of B do

    match the records in PA and PB ...

Join LRU

Clock	Block Read	Frame #1	Frame #2	Frame #3
0	PA1	*PA1 (pinned) (0)*		
1	PB1	PA1 (pinned) (0)	*PB1 (1)*	
2	PB2	PA1 (pinned) (0)	PB1 (1)	*PB2 (2)*
3	PB3	PA1 (pinned) (0)	*PB3 (3)*	PB2 (2)
4		PA1 (unpinned) (4)	PB3 (3)	PB2 (2)
5	PA2	PA1 (4)	PB3 (3)	*PA2 (5) (pinned)*
6	PB1	PA1 (4)	*PB1 (6)*	PA2 (5) (pinned)
7	PB2	*PB2 (7)*	PB1 (6)	PA2 (5) (pinned)
8	PB3	PB2 (7)	*PB3 (8)*	PA2 (5) (pinned)

Unpin PA1, which is the most recently used page.

Join (Continue) LRU

Clock	Block Read	Frame #1	Frame #2	Frame #3
8	PB3	PB2 (7)	*PB3 (8)*	PA2 (5) (pinned)
9		PB2 (7)	PB3 (8)	PA2 (9) (unpinned)
10	PA3	PA3 (10) (pinned)	PB3 (8)	PA2 (9)
11	PB1	PA3 (10) (pinned)	*PB1 (11)*	PA2 (9)
12	PB2	PA3 (10) (pinned)	PB1 (11)	*PB2 (12)*
13	PB3	PA3 (10) (pinned)	*PB3 (13)*	PB2 (12)
14		PA3 (14) (unpinned)	PB3 (13)	PB2 (12)

Using **LRU**, every scan of a page will cost a page miss.so we need to re-read to get 2nd record - so IOs is #pages in A * #pages in B. However, **MRU** will do much better.

Join MRU

Clock	Block Read	Frame #1	Frame #2	Frame #3
0	PA1	*PA1 (pinned) (0)*		
1	PB1	PA1 (pinned) (0)	*PB1 (1)*	
2	PB2	PA1 (pinned) (0)	PB1 (1)	*PB2 (2)*
3	PB3	PA1 (pinned) (0)	PB1 (1)	*PB3 (3)*
4		PA1 (unpinned) (4)	PB1 (1)	PB3 (3)
5	PA2	*PA2 (5)*	PB1 (1)	PB3 (3)
6	PB1	PA2 (5) (pinned)	PB1 (6) (page hit)	PB3 (3)
7	PB2	PA2 (5) (pinned)	*PB2 (7)*	PB3 (3)
8	PB3	PA2 (5) (pinned)	PB2 (7)	PB3 (8) (page hit)

- Unpin PA1, which is the most recently used page.
- Read PA2 to replace PA1.

Join (Continue) MRU

Clock	Block Read	Frame #1	Frame #2	Frame #3
8	PB3	PA2 (5) (pinned)	PB2 (7)	PB3 (8) (page hit)
9		PA2 (9) (unpinned)	PB2 (7)	PB3 (8)
10	PA3	*PA3 (10) (pinned)*	PB2 (7)	PB3 (8)
11	PB1	PA3 (10) (pinned)	PB2 (7)	*PB1 (11)*
12	PB2	PA3 (10) (pinned)	PB2 (12) (page hit)	PB1 (11)
13	PB3	PA3 (10) (pinned)	*PB3 (13)*	PB1 (11)
14		PA3 (14) (unpinned)	PB3 (13)	PB2 (11)

Using **MRU**, we have #frame -1 page hits in 2nd scan

More Discussion

- MRU isn’t always better either
- Example: Root vs. lower-level nodes of index, lots of pages
  - MRU keeps low-level from old searches, may discard root
  - LRU more likely to keep root

Want to know more, see

Hong-Tai Chou and David J. DeWitt. An Evaluation of Buffer Management Strategies for Relational Database Systems.

http://www.vldb.org/conf/1985/P127.PDF

Thank You!